Development Plan
RoCam

Team #3, SpaceY
Zifan Si
Jianqing Liu
Mike Chen
Xiaotian Lou

Table 1: Revision History

Date	**Developer(s)**	**Change**
Sept. 10, 2025	Mike Chen	Proof of Concept Plan and Expected Technology
Sept. 13, 2025	Jianqing Liu	Team Member Roles and Communication Plan
Sept. 13, 2025	Jianqing Liu	Expected Technology and Coding Standard

1 Confidential Information?

2 IP to Protect

3 Copyright License

4 Team Meeting Plan

5 Team Communication Plan

- **Discord**: Used for general team communication, informal discussions, quick updates, and coordination of meetings. Team members will use Discord for day-to-day conversations, sharing progress updates, asking general questions, and maintaining team cohesion.
- **GitHub**: Used for all code-related discussions and project management. GitHub Issues will be used to track bugs, feature requests, and tasks. GitHub Pull Requests will be used for code reviews and discussions related to specific code changes. Any technical discussions related to implementation details, code quality, or specific issues will be conducted through GitHub's commenting system on the relevant issue or pull request.

6 Team Member Roles

- **Project Manager**: Oversees project timeline, coordinates tasks between team members, manages deliverables and deadlines, and serves as primary point of contact with supervisor and stakeholders.
- **Meeting Chair**: Leads team meetings, prepares agendas, ensures discussions stay on track, facilitates decision-making, and manages meeting time effectively.
- **Notetaker**: Records meeting minutes, tracks action items and decisions, maintains documentation of team discussions, and distributes meeting summaries to all members.
- **Quality Assurance**: Reviews code, documentation, and deliverables for quality and consistency, conducts testing and validation, ensures adherence to coding standards and project requirements, and manages the review process for all team outputs.

7 Continuous Integration Plan

use Black for formatting and pylint and Ruff for static analysis;
use pytest + coverage for tests ($\geq 50\%$) on Python 3.9–3.11 across Linux, macOS, and Windows;
perform SAST with Bandit; all of the above are enforced via GitHub Actions.

linter tools: pylint and ruff
unit testing: pytest
code coverage: coverage
code security: bandit

- How will you be using git, including branches, pull request, etc.?
- How will you be managing issues, including template issues, issue classification, etc.?
- Use of CI/CD

8 Project Decomposition and Scheduling

- How will you be using GitHub projects?
- Include a link to your GitHub project

9 Proof of Concept Demonstration Plan

The following are planned steps of POC:

1. Have initial image acquired from a camera, where the target is stationary.
2. Activate tracing mode of our system, it will segment and detect multiple moving objects in the image.
    - The segmentation and moving object detection will be done using Computer Vision techniques.
    - The Computer Vision model will be deployed on a Jetson Nano.
3. user select a stationary or moving object as the target.
4. As the target moves, the system will keep it at the center of the image for smooth tracking.
    - The system will be able to handle occlusion and loss of target.
    - user can manually re-select the target if needed.
    - the real time control of the camera will be done using a STM32 microcontroller.

The following is a list of primary risks to consider from the POC:

1. The computer vision system may not be able to process images at a fast enough rate.
    - If this occurs, we will try to optimize the existing model, consider using a more powerful board, lower frame rate, or consider using traditional algorithmic approach and technique, where it detects motion by comparing image pixel wise.
    - We also reserve the option of use models trained for specific set of objects to increase the speed of the model.
2. The STM32 may not be able to deliver the real time control to the camera.
    - If this occurs, we will consider using a more powerful microcontroller, or using a different control technique.
3. The Integration of the frontend, backend, computer vision model, and microcontroller may be more difficult than expected.
    - If this occurs, we will consider using a more powerful integrated micro computer or deploy via cloud computation, redesign our control flow, to make integration easier.

Other smaller risks to consider:

1. UI Useability issues: The user interface may not be intuitive or easy to use, leading to user frustration or errors.
    - Potential Solution: Conduct user testing and gather feedback to improve the interface design.

10 Expected Technology

- Motion Control
  - STM32 Microcontroller
  - Rust Programming Language
  - Linter: rust-clippy
  - Unit Testing: Rust built-in
  - Code Coverage: grcov
  - Hardware Abstraction Layer: embassy-rs
  - Debugging: probe-rs
- Computer Vision
  - Nvidia Jetson
  - Nvidia Jetpack
  - Language: Python
  - Libraries: OpenCV, NumPy, Matplotlib, Torch
  - Open Source Models: Ultralytics YOLO, SAM, various ViTs
  - Linter: pylint
- Web App
  - Web Server: Flask
  - React
  - Typescript
  - Linter: eslint
  - End-to-end Testing: Cypress
- All the above will use GitHub Actions for CI
- Development Tools
  - VSCode
  - PyCharm
  - Git
  - Github

11 Coding Standard

- Rust: The Power of 10 Rules
- Python: Follow PEP 8 / PEP 257 / PEP 484; use Black for formatting and pylint and Ruff for static analysis;
  use pytest + coverage for tests ($\geq 50\%$) on Python 3.9–3.11 across Linux, macOS, and Windows; perform SAST with Bandit; all of the above are enforced via GitHub Actions.

  If the project matures, gradually raise the coverage threshold from 50% to 70%–80% (by increasing the threshold step by step in CI).

- Typescript: typescript-eslint

Appendix — Reflection

The purpose of reflection questions is to give you a chance to assess your own learning and that of your group as a whole, and to find ways to improve in the future. Reflection is an important part of the learning process. Reflection is also an essential component of a successful software development process.

Reflections are most interesting and useful when they're honest, even if the stories they tell are imperfect. You will be marked based on your depth of thought and analysis, and not based on the content of the reflections themselves. Thus, for full marks we encourage you to answer openly and honestly and to avoid simply writing "what you think the evaluator wants to hear."

Please answer the following questions. Some questions can be answered on the team level, but where appropriate, each team member should write their own response:

1. Why is it important to create a development plan prior to starting the project?
2. In your opinion, what are the advantages and disadvantages of using CI/CD?
3. What disagreements did your group have in this deliverable, if any, and how did you resolve them?

Appendix — Team Charter

External Goals

Attendance

Expectations

Acceptable Excuse

In Case of Emergency

Accountability and Teamwork

Quality

Attitude

Stay on Track

Team Building

Decision Making